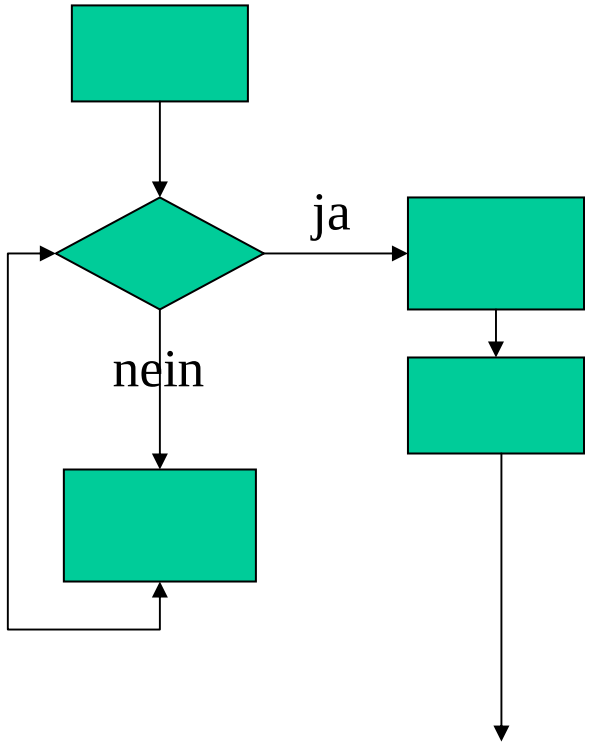
h_da
HOCHSCHULE DARMSTADT
UNIVERSITY OF APPLIED SCIENCES

5. Elementare Befehle und Struktogramme

Programmablauf

Beschreibung des Programmablaufs mittel grafischer Symbole

Beispiel : Flussdiagramme



Besser : Struktogramme

h_da
HOCHSCHULE DARMSTADT
UNIVERSITY OF APPLIED SCIENCES

Elementare Befehle

Befehlstypen

Zuweisung

Verzweigung

Schleife

5.1 Zuweisungen

Einfache Zuweisung

Beispiel flaeche=r*r*3.14159;

Struktogramm

flaeche = r*r*3.14159

oder

flaeche = r^2*3.14159

Inhalte sind beliebig zu beschreiben

Einfache Zuweisung

Aufbau

Variable = Ausdruck

Allgemein
L-Wert Zuweisungsoperator R-Wert
(L-value) (R-value)

L-Wert bezeichnet immer eine Speicherstelle zur Speicherung des Ergebnisses, z.B. ein Variablenname

R-Werte können auch Ausdrücke sein wie x*y+1

Zusammengesetzte Zuweisung

Anstelle von	i=i+1;	i+=1;
	d=d*(r+2);	d*=r+2;
Prinzip	v=v op r;	v op= r;

Beispiele += -= *= /= %=

5.2 Verzweigungen

Typen

if

if else

else-if Kette

switch

Auswahloperator

if

```
if ( boolscher Ausdruck)
        Anweisung1
Anweisung2
```

Anweisung : Befehl;

{Befehl;Befehl;.....;Befehl;}

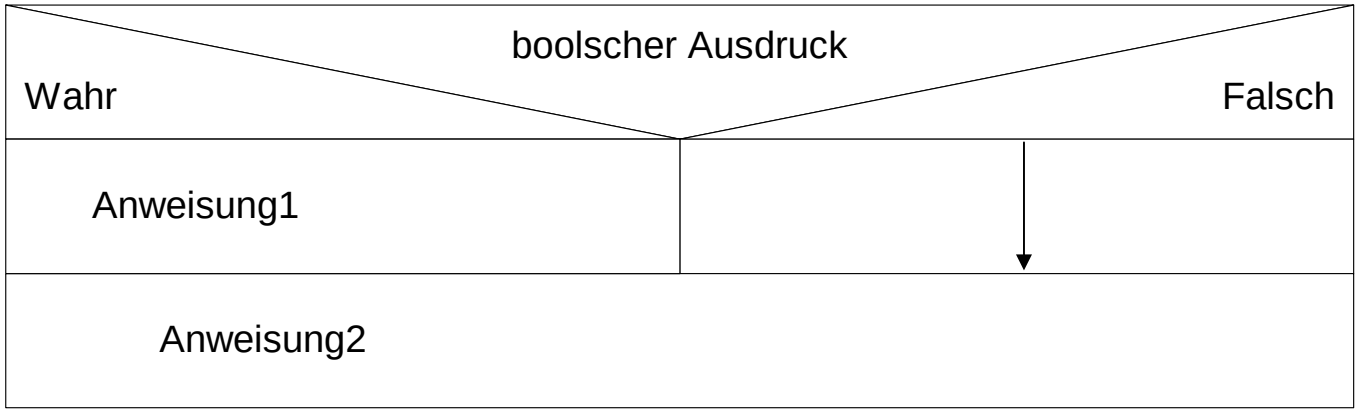
Wirkungsweise

Auswertung des boolschen Ausdrucks
falls true : Führe Anweisung1 aus, dann Anweisung2
falls false: Führe Anweisung2 aus

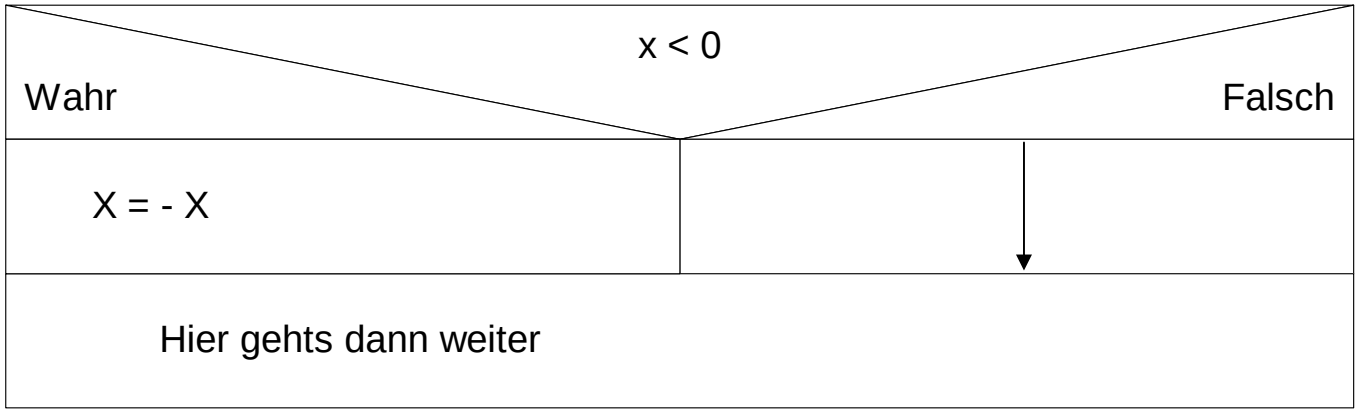
Beispiel

```
if ( x < 0 )
        x = - x;
// jetzt ist x immer >= 0
```

Struktogramm



Beispiel



if-else

```
if ( boolscher Ausdruck)
    Anweisung1
else
    Anweisung2
Anweisung3
```

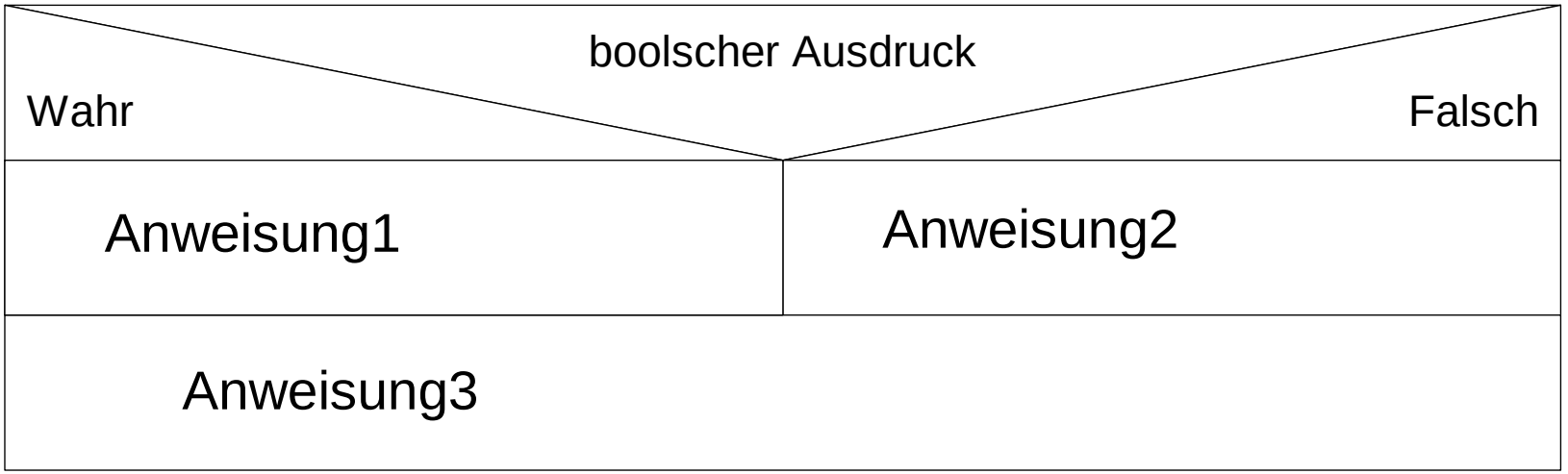
Wirkungsweise

- Auswertung des boolschen Ausdrucks
- falls true : Führe Anweisung1 aus, dann Anweisung3
- falls false: Führe Anweisung2 aus, dann Anweisung3

Beispiel

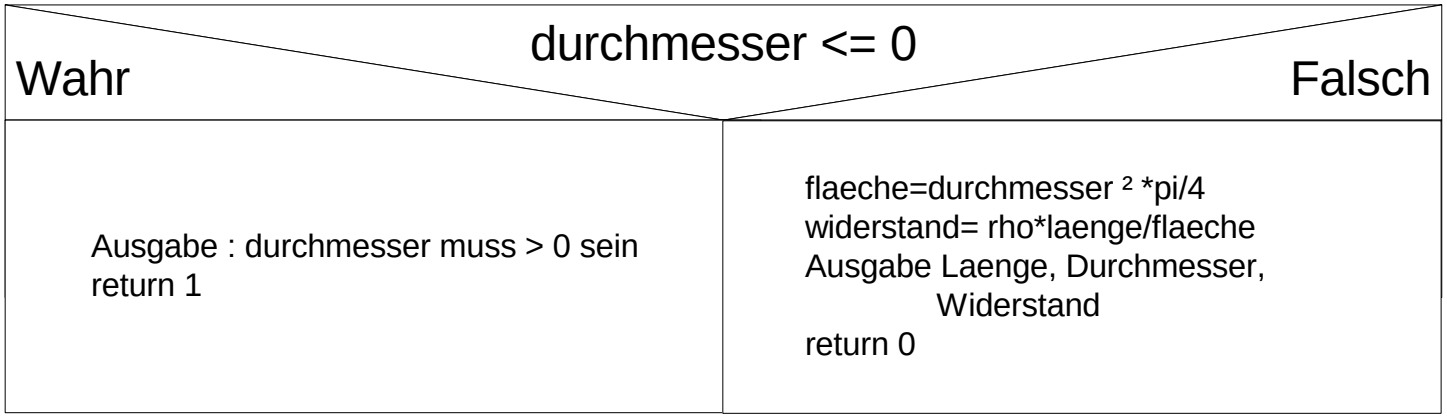
```
if ( durchmesser <= 0 )                    // unzulässig
        {
                cout << " Durchmesser muss > 0 sein ";
                return 1;
        }
        else                                                    // Eingabedaten in Ordnung
        {
                flaeche=pow(durchmesser,2)*pi/4;
                widerstand= rho*laenge/flaeche;
//              Ausgabe
                cout << " Laenge \t" << laenge << " m" << endl;
                cout << " Durchmesser \t" << durchmesser <<" mm" << endl;
                cout << " Widerstand \t" <<widerstand <<" Ohm"<<endl;
                return 0;
        }
```

Struktogramm



h_da
HOCHSCHULE DARMSTADT
UNIVERSITY OF APPLIED SCIENCES

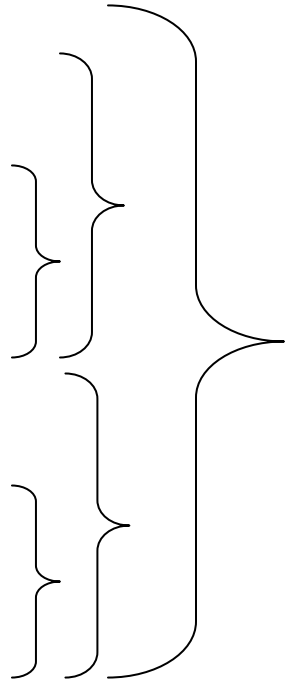
Beispiel



Verschachtelung

```
// Verschachtelung von if else
#include <iostream>
using namespace std;
void main()
{
        int i,j,k;
        cin >> i >> j >> k;
        if ( i < j )
                if ( j < k )
                        cout << i << j << k;
                else
                        if ( i < k )
                                cout << i << k << j;
                        else
                                cout << k << i << j;
        else
                if ( i < k )
                        cout << j << i << k;
                else
                        if ( j < k )
                                cout << j << k << i;
                        else
                                cout << k << j << i;
}
```

if-else Befehle müssen immer vollständig im if oder else Teil eines „übergeordneten“ if-else Befehls enthalten sein



Verschachtelung/2

```
// Werte Stromimpuls
#include <iostream>
using namespace std;
void main()
{
        double t;
        cin >> t ;
        if ( t < 0 )
                cout << " Wert ist " << 0  <<endl;
        else
                if ( t < 1 )
                        cout << " Wert ist " << t <<endl;
                else
                        if ( t < 2 )
                                cout << " Wert ist " << 2-t <<endl;
                        else
                                cout << " Wert ist " << 0 <<endl;

}
```

Alternative

```
// Werte Stromimpuls
#include <iostream>
using namespace std;
void main()
{
        double t;
        cin >> t ;
        if ( t < 0 )
                cout << " Wert ist " << 0  <<endl;
        else if ( t < 1 )
                cout << " Wert ist " << t <<endl;
        else if ( t < 2 )
                cout << " Wert ist " << 2-t <<endl;
        else
                cout << " Wert ist " << 0 <<endl;

}
```

else if Kette

```
if ( boolscher Ausdruck1)
        Anweisung1
else if (booscher Ausdruck2)
        Anweisung2
else if ....

else
        Anweisungm
Anweisungn
```

Wirkungsweise

Auswertung von boolscher Ausdruck1

falls true : Führe Anweisung1 aus,
dann Anweisungn

falls false:

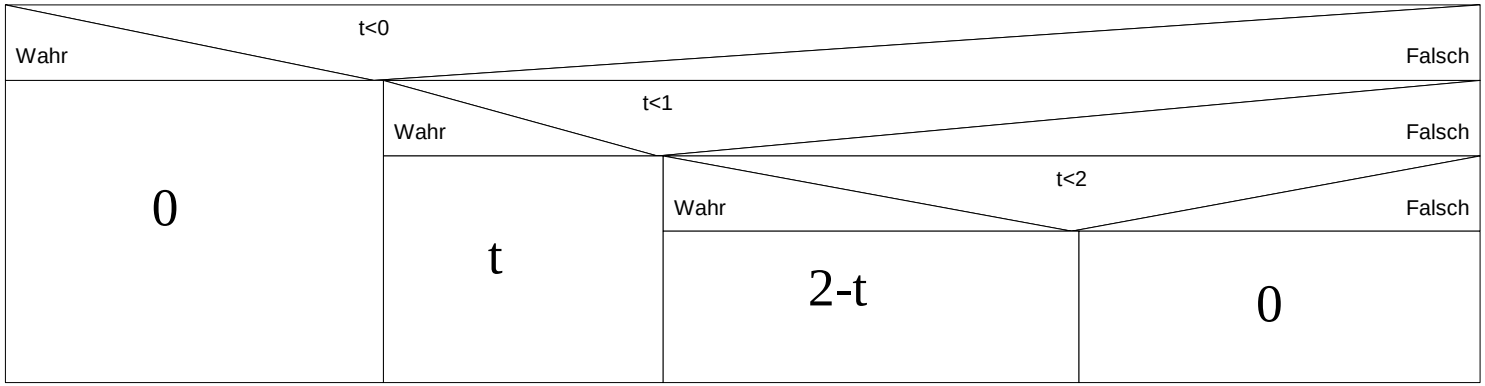
Auswertung von boolscher Ausdruck2

falls true : Führe Anweisung2 aus,
dann Anweisungn

falls false:

.....

Struktogramm



Switch/Beispiel

```
// Kleinrechner/switch
#include <iostream>
using namespace std;
void main()
{
        int i,j;
        char op;
        cin >> i >> op >> j;
        switch (op)
        {
                case '+' :
                        cout << i << op << j <<" = " << i+j << endl;
                        break;
                case '-' :
                        cout << i << op << j <<" = " << i-j << endl;
                        break;
                case '*' :
                        cout << i << op << j <<" = " << i*j << endl;
                        break;
                case '/' :
                        cout << i << op << j <<" = " << i/j << endl;
                        break;
                case '%' :
                        cout << i << op << j <<" = " << i%j << endl;
                        break;
                default :
                        cout << " unzulässiger Operator " << endl;
        }
}
```

switch

```
switch ( ganzzahliger Ausdruck )
{
        case konstante1 :
                Befehl;Befehl;...Befehl;
                break;
        case konstante2 :
                Befehl;Befehl;...Befehl;
                break;
        .....

        default :
                Befehl;Befehl;...Befehl;
}
```

Kann weggelassen werden
Nicht empfehlenswert !!!!!

switch

Wirkungsweise

Auswertung des Ausdrucks (muss ganzzahliger Typ sein)

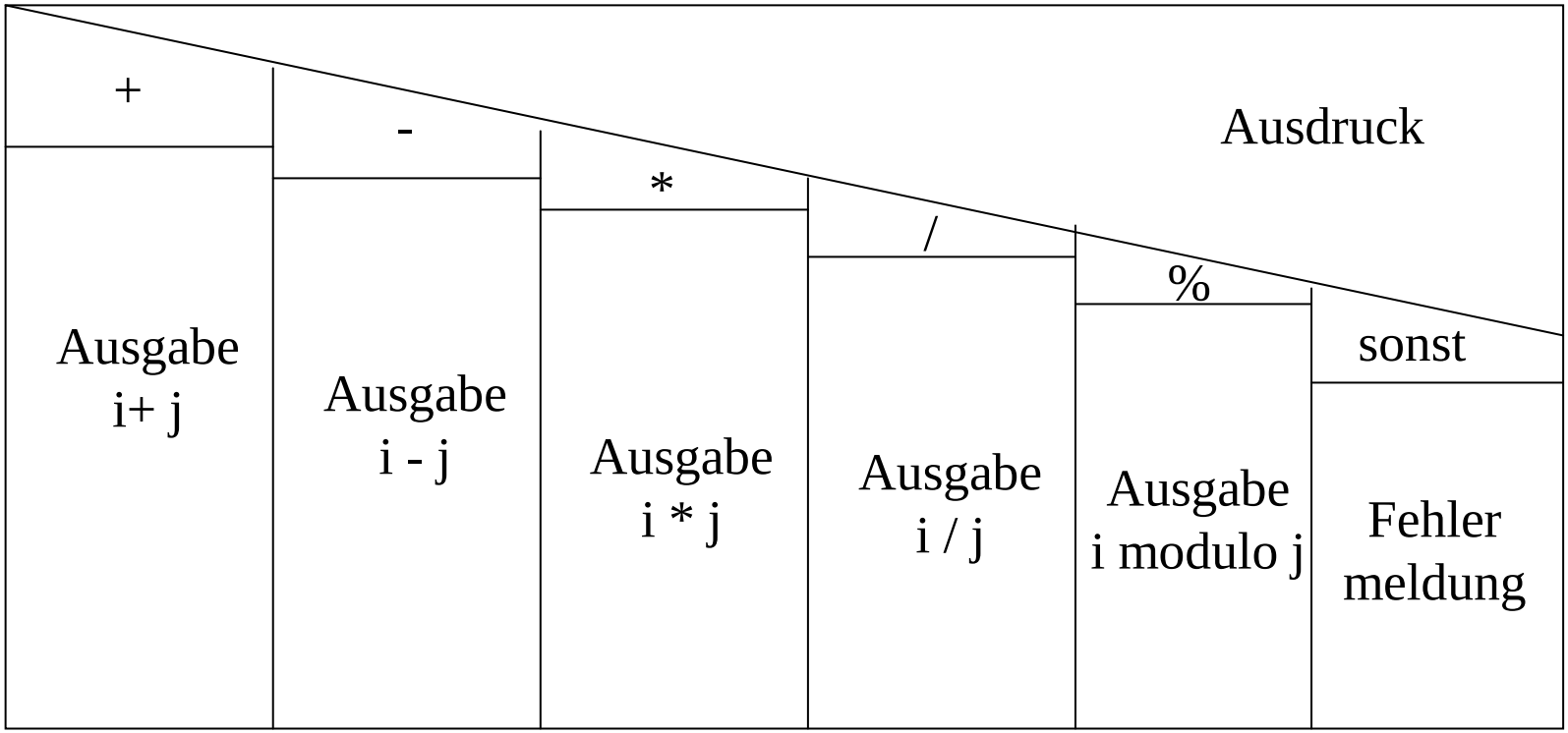
Vergleich mit den ganzzahligen Konstanten (alle verschieden)

Bei Übereinstimmung mit einer Konstanten->Verzweigung zur case – Marke.
Ausführung der Befehle bis zum break, dann Verzweigung zum Ende
(nach dem switch).

Fehlt der break-Befehl wird mit dem darauf folgenden Befehl fortgesetzt !!!!!

Stimmt der Wert mit keiner Konstanten überein wird bei default fortgesetzt,
Sofern vorhanden (wenn nicht, findet keine Aktion statt).

Struktogramm



Gegenüberstellung

Die else-if Kette ist universeller als switch. Sie kann mit beliebigen booleschen Ausdrücken formuliert werden.

Switch ist beschränkt auf den Vergleich ganzzahliger Ausdrücke (u.a. Aufzählungstyp !!), ist aber übersichtlicher, insbesondere wenn es viele Fälle sind.

Auswahloperator

Beispiel

Anstelle von

```
if ( x >= 0 )
        y = x;
else
        y = -x;
```

kann auch

```
y = (x>=0)? x : -x;
```

mit dem Auswahloperator ? verwendet werden (bedingte Bewertung).

Auswahloperator

Form :

(boolscher Ausdruck) ? Ausdruck1: Ausdruck2;

Wirkung :

Wenn der boolsche Ausdruck wahr ist, wird Ausdruck1 verwendet, sonst Ausdruck2.

5.3 Schleifen

while-Schleife

do while Schleife

for Schleife

-> Hinweis auf Entwicklung

Modellierung von while-Schleifen

1) Identifizierung der **Wiederholungs**bedingung (WB) der Schleife

   (ev. Negation einer Abbruchbedingung)

2) Ermittlung, ob die Schleife immer mindestens einmal durchlaufen wird
   falls ja: do while
   andernfalls: while

3) Identifizierung aller Befehle, die wiederholt werden wie z.B.:
   nächste Daten lesen – neue Summe berechnen – zählen – ...

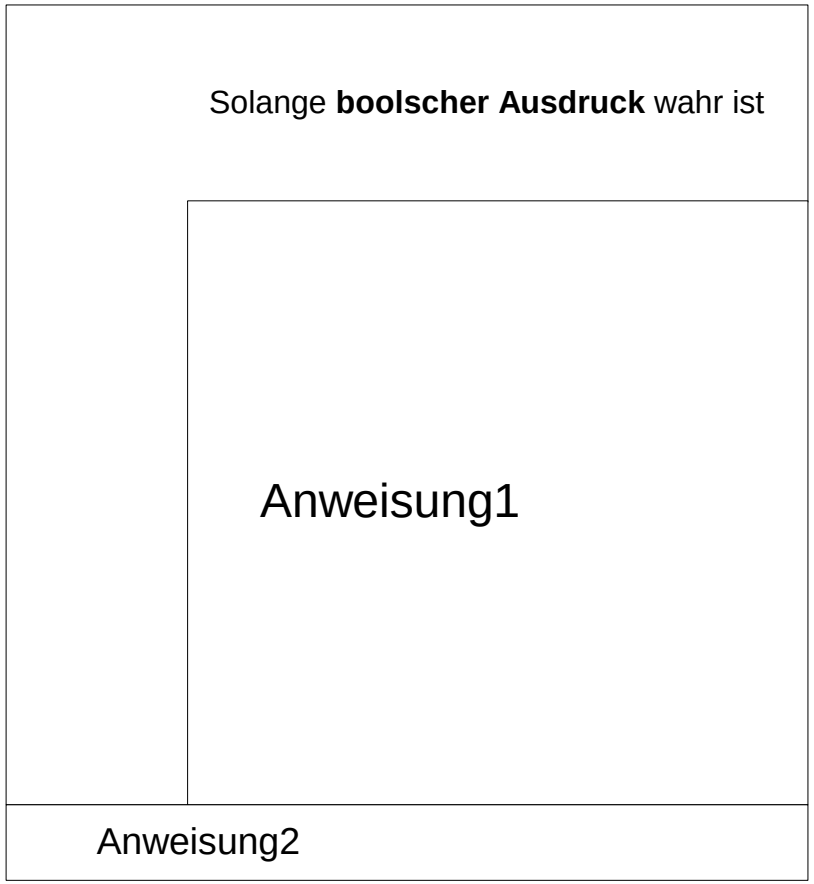
4) Existenz von Befehlen unter 3) prüfen, welche WB ändern.
   Ansonsten ist es eine unendliche Schleife.

while Schleife

while (boolescher Ausdruck)
    Anweisung1
Anweisung2

Wirkung
    (*) Werte boolschen Ausdruck aus
        falls true : Führe Anweisung1 aus
            Mache weiter bei (*)
        falls false: Führe Anweisung2 aus

Struktogramm



while Schleife/Beispiel

In einem Programm sollen positive ganze Zahlen eingelesen und deren Durchschnitt berechnet werden.

Das Programm endet, wenn eine negative Zahl oder 0 eingegeben wird.

Danach sind Durchschnitt und Anzahl der gelesenen Zahlen am Bildschirm auszugeben.

Modellierung

1- Wiederholungsbedingung *gelesene Zahl>0*
2- es muss ev. nichts getan werden -> *while-Schleife*
3- zu wiederholende Befehle: *zählen*, *summieren*, *lesen*
4- der Lesebefehl beeinflusst die Wiederholungsbedingung

Beispiel

```
// Lesen von positiven Zahlen
#include <iostream>
using namespace std;
void main()
{
        int anzahl=0,eingabezahl;
        float summe=0.0;
// Erste Zahl eingeben
        cout << " Bitte positive Zahl eingeben " << endl;
        cout << " Abbruch mit 0 " << endl;
        cin >> eingabezahl;
//Schleife
        while (eingabezahl>0)
        {
                ++anzahl;
                summe+=eingabezahl;
                cout << " Bitte naechste Zahl eingeben/Abbruch mit 0 " << endl;
                cin >> eingabezahl;
        }
//Ergebnis ausgeben
        cout << " Der Durchschnitt ist : "      << summe/anzahl << endl;
        cout << " Es wurden "<<anzahl<<" Zahlen gelesen"<<endl;
}
```

Welche Testfälle sollte man wählen?

-> Testplan

Welche Schwächen hat dieses Programm?

Beispiel

```
 Bitte positive Zahl eingeben
 Abbruch mit 0
1
 Bitte naechste Zahl eingeben/Abbruch mit 0
2
 Bitte naechste Zahl eingeben/Abbruch mit 0
2
 Bitte naechste Zahl eingeben/Abbruch mit 0
1
 Bitte naechste Zahl eingeben/Abbruch mit 0
2
 Bitte naechste Zahl eingeben/Abbruch mit 0
0
 Der Durchschnitt ist : 1.6
 Es wurden 5 Zahlen gelesen
Press any key to continue
```

```
 Bitte positive Zahl eingeben
 Abbruch mit 0
10
 Bitte naechste Zahl eingeben/Abbruch mit 0
0
 Der Durchschnitt ist : 10
 Es wurden 1 Zahlen gelesen
Press any key to continue_
```

```
 Bitte positive Zahl eingeben
 Abbruch mit 0
0
 Der Durchschnitt ist : -1.#IND
 Es wurden 0 Zahlen gelesen
Press any key to continue_
```

do while Schleife

do
Anweisung1
while (boolescher Ausdruck)
Anweisung2

Wirkung

(*) Führe Anweisung1 aus
Werte boolschen Ausdruck aus
falls true : Mache weiter bei (*)
falls false: Führe Anweisung2 aus

In einer do while Schleife wird die Anweisung in der Schleife mindestens 1 mal ausgeführt !!!!

Struktogramm

Beispiel

Es soll näherungsweise die Wurzel einer Zahl a > 0 mit folgendem Verfahren berechnet werden :

$x_0 = a \qquad x_{k+1} = (x_k + a/ x_k) /2$ für k=0,1,2,...

Man beende das Verfahren, wenn $|x_k * x_k - a| < {}_{10}$-10 gilt

Modellierung

1- Wiederholungsbedingung *Betrag*>= $_{10}$-10
2- x_1 sollte berechnet werden -> *do while-Schleife*
3- zu wiederholende Befehle: *Formel*
4- die Formel beeinflusst die Wiederholungsbedingung

h_da
HOCHSCHULE DARMSTADT
UNIVERSITY OF APPLIED SCIENCES

Beispiel/Ergebnis

```
// Wurzel
#include <iostream>
#include <cmath>
using namespace std;
void main()
{

                double x,a;
// Daten eingeben
                cout << " Bitte a>0 eingeben " << endl;
                cin >> a;
//Algorithmus mit Plausubilitätskontrolle
                if ( a> 0 )
                                {
                                x = a; // Startwert
                                do
                                {
                                                x = (x+a/x)/2;//Formel
                                }
                                while (fabs (x*x-a) >=1.0e-10 );
//                Ausgabe
                                cout << " Wurzel(" << a <<") = " << x << endl;
                                }
                else
                                cout << " a muss > 0 sein "     << endl;
}
```

Testfälle

```
 Bitte a>0 eingeben
2
 Wurzel(2) = 1.41421
Press any key to continue
```

```
 Bitte a>0 eingeben
1
 Wurzel(1) = 1
Press any key to continue_
```

```
 Bitte a>0 eingeben
-2
 a muss > 0 sein
Press any key to continue
```

```
 Bitte a>0 eingeben
0
 a muss > 0 sein
Press any key to continue_
```

for Schleife / Beispiel für Zählschleife

```
for ( i=1 ; i<n ; i+=1 )
        Anweisung1
Anweisung2
```

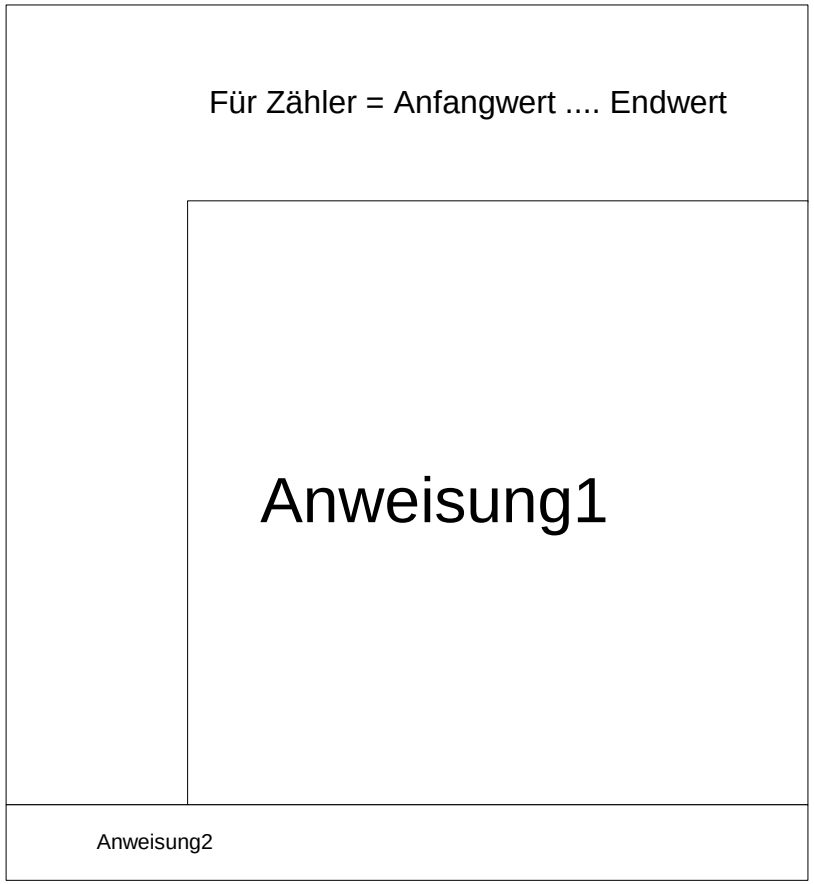
Wirkung

Führe den ersten Befehl (hier z.B. i=1) aus

(*) Werte den boolschen Ausdruck (hier z.B. i<n) aus
Falls true : Führe Anweisung1 aus
Führe die letzte Anweisung (hier z.B. i+=1) aus
Weiter bei (*)

Falls false: Weiter mit dem ersten Befehl nach der Schleife
Anweisung2

(vereinfachtes) Struktogramm



Beispiel

Man berechne die Summe der ersten n natürlichen Zahlen.
n ist einzulesen.

```
// Summe natuerlicher Zahlen
#include <iostream>
using namespace std;
void main()
{
        int i,n,summe=0;
//eingabe
        cout << " Bitte n eingeben " << endl;
        cin >>n;
//Summation
        for (i=1;i<=n;i++)
                summe+=i;
//Ergebnisausgabe
        cout << " Die Summe der ersten " << n <<" natuerlichen Zahlen ist ";
        cout << summe << endl;
}
```

```
 Bitte n eingeben
10
 Die Summe der ersten 10 natuerlichen Zahlen ist 55
Press any key to continue
```

```
 Bitte n eingeben
1
 Die Summe der ersten 1 natuerlichen Zahlen ist 1
Press any key to continue
```

```
 Bitte n eingeben
0
 Die Summe der ersten 0 natuerlichen Zahlen ist 0
Press any key to continue_
```

for-Schleife allgemein

for (Anweisung1;boolscher Ausdruck;Anweisung3)
    Anweisung2
Anweisung4

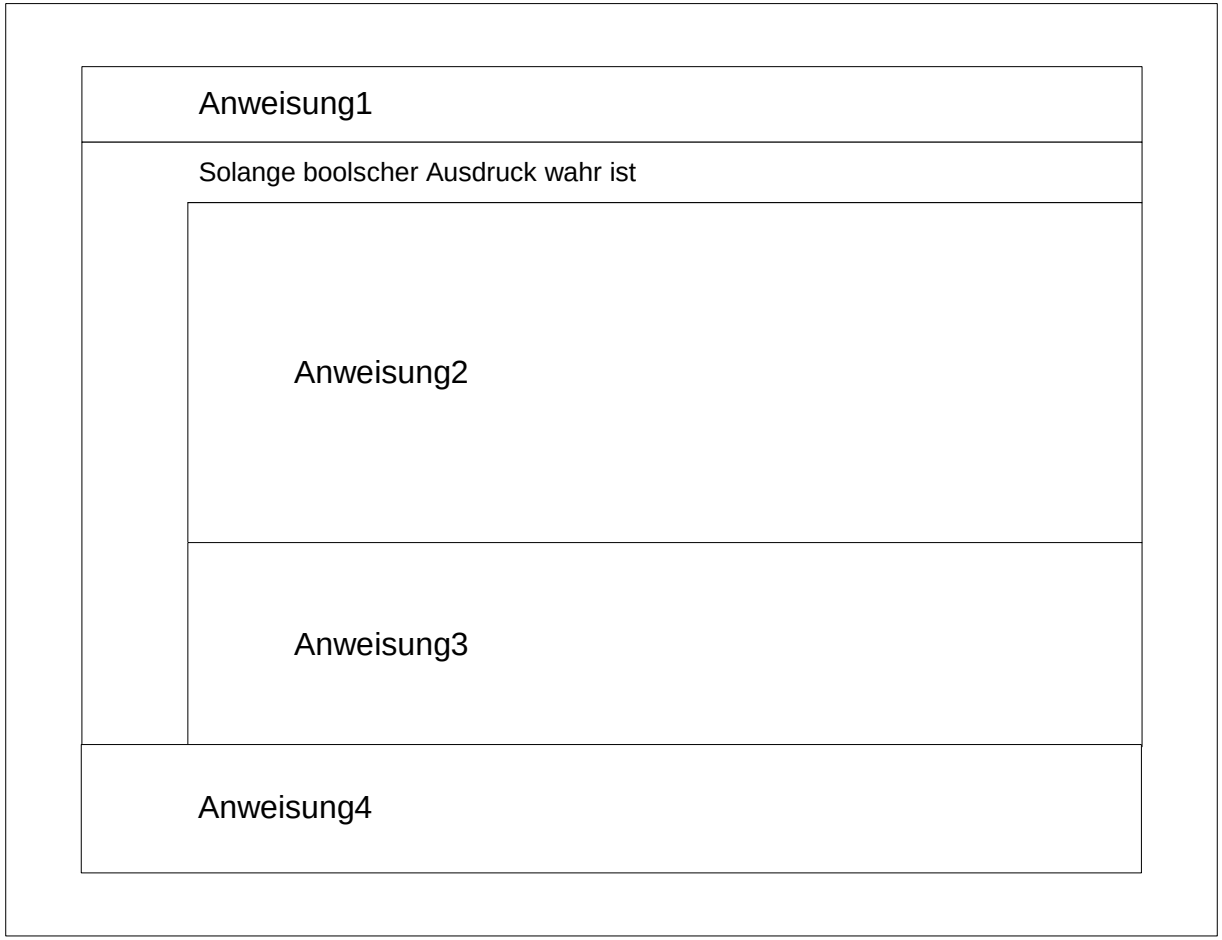
Wirkung

Führe Anweisung1(=**Initialisierung**) aus
(ev. mehrere durch Komma getrennt)

(*) Werte den boolschen Ausdruck aus
Falls true : Führe Anweisung2 aus
Führe Anweisung3 (=**Reinitialisierung**) aus
(ev. mehrere durch Komma getrennt)
Weiter bei (*)

Falls false: Weiter mit dem ersten Befehl nach der Schleife
Anweisung4

Struktogramm



For-Schleife/Beispiele

```
for (i=0 ;  i<n ; i++)
        {....}

for (i=0 ;  i<n ; i+=2)
        {....}

for (k=m ;  k>0 ; k--)
        {....}


for (i=1,k=100;  i<n && k > 0; i++,k--)
        {....}


for (Datei oeffnen, hole erste Daten ; kein Dateiende ;  lese naechste Daten)
        {verarbeite Daten}
```

5.4 Sonstige Befehle

break	Verzweigung ans Ende eines Switch-Befehls oder einer Schleife. Weiter mit dem darauf folgenden Befehl.
continue	Beginne mit der nächsten Wiederholung einer Schleife while/do-while : Wiederholungsbedingung for : Reinitialisierung
goto marke	Verzweigung zu einer Marke. Eine Marke ist ein Name gefolgt von : Eine Marke kann vor jedem Befehl stehen. Tipp : goto vermeiden (erhöht Intransparenz).